ASRIC: ADAPTIVE SPARSE ROUTING FOR LEARNED IMAGE COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned image codecs often allocate transform computation and entropy-model capacity uniformly across heterogeneous content. As such, low-complexity regions incur the same dense processing as structurally complex regions, while a fixed pool of entropy-model features accommodates diverse conditional latent distributions. This allocation can incur redundant computation on unnecessary transform updates and limit the specialization of probability estimates to local context, resulting in limited compression performance under a fixed computational budget. In this paper, we propose Adaptive Sparse Routing for Learned Image Compression (ASRIC) to match transform computation and entropy-model capacity to content through two complementary mechanisms. Within the transforms, Spatially Adaptive Mamba blocks are introduced to select spatial tokens and scan directions for recurrent processing based on the Mamba module, while coordinate-dependent state transitions preserve the geometric separation between selected tokens. For entropy modeling, decoder-available context selects Top-K prototype banks, providing specialized statistical features for each prediction without transmitting bank indices. Experimentally, ASRIC achieves BD-Rate reductions of 19.15%, 18.67%, and 20.37% relative to VTM-9.1 on Kodak, CLIC, and Tecnick, respectively. Compared with CMIC, the strongest learned baseline in our comparison, ASRIC improves the VTM-referenced BD-Rate by 0.94, 0.81, and 0.32 percentage points across the three benchmarks, while requiring 4.2% fewer FLOPs and reducing encoding and decoding time by 5.6% and 16.3%, respectively.

1 INTRODUCTION

Learned image compression (LIC) has improved rate–distortion (RD) performance through expressive transforms and conditional entropy models (Ballé et al., 2018; Cheng et al., 2020; Li et al., 2024; 2025). Resources are commonly allocated uniformly: transforms apply the same long-range processing at each feature location, while entropy models use a shared pool of statistical capacity for latent probability estimation. This can incur redundant transform updates with limited RD benefit and increase bitrate when the shared predictive features inadequately represent the conditional latent distributions, limiting compression performance under a fixed computational budget.

On the transform side, Mamba-based state-space models (SSMs) combine long-range modeling with linear complexity in sequence length (Gu & Dao, 2024). MambaIC uses multidirectional scans to aggregate long-range information (Zeng et al., 2025); CMIC adapts the scan order and introduces global-prior prompts (Chen et al., 2026b). These SSM codecs still process every token, including locations where local features may already provide sufficient support, so additional recurrent updates can consume computation with limited RD benefit. Under a fixed budget, these updates restrict the computation available for more demanding structures. Selecting tokens and directions can reduce this work, but sparse propagation should retain the spatial separation lost when selected tokens are packed into a shorter sequence. On the entropy side, HPCM accumulates context across scales and coding steps (Li et al., 2025), while DCAE queries a learned dictionary to exploit structures recurring across training images (Lu et al., 2025). These methods already produce context-dependent predictions. The remaining issue is how to make a richer set of statistical features available without increasing the work for every prediction. DCAE's attention weights are content dependent, yet each query accesses the full dictionary; enlarging it increases retrieval cost at every location. Conversely,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

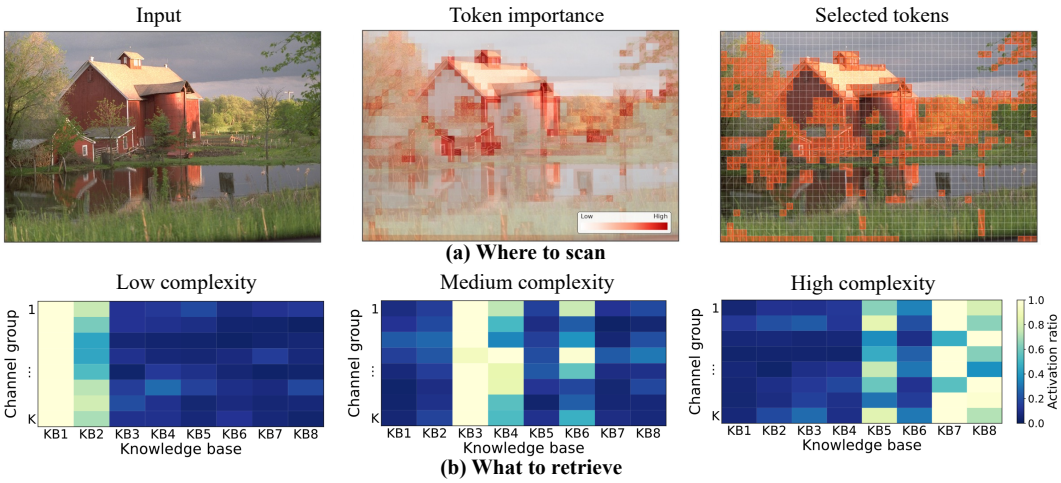


Figure 1: Content-dependent routing in ASRIC. (a) Token Importance Estimator (TIE) scores and hard selections on Kodak `kodim22` at $\lambda = 0.013$ show nonuniform spatial execution. (b) Normalized Top-K activation frequencies for eight knowledge bases (KBs), aggregated over all 24 Kodak images, show that relative bank usage varies across complexity bins. Fixed high-frequency-energy quantiles define these bins independently of the router.

restricting the dictionary to fit a fixed retrieval budget can limit the range of structures represented by its features and leave conditional distributions less accurately modeled. When the predicted distribution differs from the true conditional distribution, the excess cross entropy increases the expected code length even for unchanged latents and reconstructions. This motivates selecting specialized statistical features from a larger memory under a bounded retrieval budget. Such selection should use already decoded context so that both codec ends obtain the same probabilities without transmitting selection indices.

Taken together, these limitations expose a common allocation problem: how to provide content-specific transform processing and probability prediction without evaluating every long-range update or dictionary entry at every location. We propose Adaptive Sparse Routing for Learned Image Compression (ASRIC) to address this joint allocation problem. Spatially Adaptive Mamba (SAM) blocks use an Adaptive Spatial Routing Module (ASRM) to predict feature-dependent token and direction scores. Recurrent computation is restricted to selected tokens and directions under compute constraints, with coordinate-dependent state transitions retaining selected tokens' feature-grid displacement. The Knowledge-Augmented Entropy Model (KAEM) uses Memory-Augmented Sparse Knowledge Routing (MASKR) to score prototype banks from decoder-available context and retrieve only from the Top-K banks, adapting predictive features without transmitting bank indices. Figure 1 illustrates how the two mechanisms allocate spatial computation and statistical capacity. Joint RD training coordinates both mechanisms, with transform decisions derived from each transform's input features and entropy decisions restricted to the hyperprior and previously decoded latents.

In summary, our contributions are as follows:

- We introduce content-adaptive token and direction selection for SSM transforms. Coordinate-dependent state transitions preserve geometric separation after compaction, and an explicit gather–reorder–scan–scatter implementation executes the selected updates.
- We introduce adaptive sparse routing for entropy modeling. Decoder-available context selects Top-K prototype banks and restricts retrieval to this subset, adapting predictive features to content without transmitting bank indices.
- Experiments on Kodak, CLIC, and Tecnick show complementary RD gains from the two mechanisms. Compared with CMIC, the strongest learned baseline in our comparison, ASRIC improves the VTM-referenced BD-Rate scores by 0.94, 0.81, and 0.32 percentage points, respectively. On CLIC, ASRIC requires 4.2% fewer FLOPs and reduces encoding and decoding time by 5.6% and 16.3%, respectively.

2 Related Work

2.1 Learned Image Compression

Learned entropy models combine hyperpriors with decoded neighbors (Minnen et al., 2018), reduce serial dependencies through checkerboard prediction and channel grouping (He et al., 2021; 2022), and aggregate long-range evidence through attention and multireference contexts (Kim et al., 2022; Qian et al., 2022; Jiang et al., 2023). Causal context adjustment (Han et al., 2024) and hierarchical progressive modeling (Li et al., 2025) further organize the evidence available for probability estimation.

Transform design has advanced in parallel through attention and mixture likelihoods (Cheng et al., 2020), mixed Transformer–convolutional architectures (Liu et al., 2023), frequency-aware operators (Li et al., 2024; Fu et al., 2024), and efficient global attention (Feng et al., 2025). Shallow decoders shift representational capacity and iterative optimization toward the encoder (Yang & Mandt, 2023). ASRM adapts the execution of a transform block: it selects locations and propagation directions for long-range processing while retaining dense local processing.

KAEM is closest to DCAE, which queries a learned dictionary through cross-attention to capture structures recurring across training images (Lu et al., 2025). KAEM partitions this dataset-level memory into KBs and selects Top-K banks from decoder-available context at each spatial position within a channel group. Its distinction lies in bank selection and a readout shared across the group's channels, rather than dictionary learning itself. Attention remains dense within activated banks; memory is fixed during coding, and bank indices are reproduced without side information.

2.2 State-Space Models and Routing

Mamba conditions state-space parameters on the input and uses a linear-time scan (Gu & Dao, 2024); structured state-space duality connects these operators to attention (Dao & Gu, 2024). Vision Mamba uses bidirectional processing (Zhu et al., 2024), and VMamba introduces cross-scanning for images (Liu et al., 2024). Input-dependent state parameters, however, do not by themselves skip token updates.

QuadMamba preserves locality through quadtree partitions (Xie et al., 2024), while MambaIRv2 connects similar pixels through semantic-guided neighboring (Guo et al., 2025). Hidden-state alignment addresses disruptions caused by token pruning (Zhan et al., 2024). ASRM combines variable-cardinality token selection, direction routing, and coordinate-dependent transitions, retaining geometric separation after sequence compaction.

For compression, MambaIC incorporates visual SSMs into transforms and entropy modeling (Zeng et al., 2025). Content-Aware Mamba (CMIC) learns content-dependent permutations (Chen et al., 2026b), making it a close comparison for adaptive SSM transforms; GLIC adapts graph neighborhoods (Chen et al., 2026a). ASRIC additionally selects which token and direction updates execute and uses a separate decoder-causal route over prototype banks.

Conditional vision models prune tokens or vary depth (Rao et al., 2021; Liang et al., 2022; Yin et al., 2022), and sparse mixture-of-experts models select parameter blocks (Riquelme et al., 2021; Zhou et al., 2022). ASRIC applies related selection principles while preserving spatial coordinates in sparse propagation and decoder reproducibility in entropy routing.

3 Method

3.1 Overview and Probabilistic Formulation

Adaptive allocation requires separate decisions for transform computation and entropy-model capacity. ASRIC combines a spatially routed transform with a prototype-based entropy model. For an image $\mathbf{x} \in \mathbb{R}^{3\times H_0\times W_0}$, the codec computes:

$$\mathbf{y} = g_a(\mathbf{x}), \quad \mathbf{z} = h_a(\mathbf{y}), \quad \hat{\mathbf{y}} = Q(\mathbf{y}), \quad \hat{\mathbf{z}} = Q(\mathbf{z}), \quad \hat{\mathbf{x}} = g_s(\hat{\mathbf{y}}), \tag{1}$$

where g_a and g_s are the analysis and synthesis transforms, h_a is the hyper-analysis transform, and Q denotes quantization. The primary latent $\mathbf{y}$ and hyperlatent $\mathbf{z}$ have quantized counterparts $\hat{\mathbf{y}}$ and

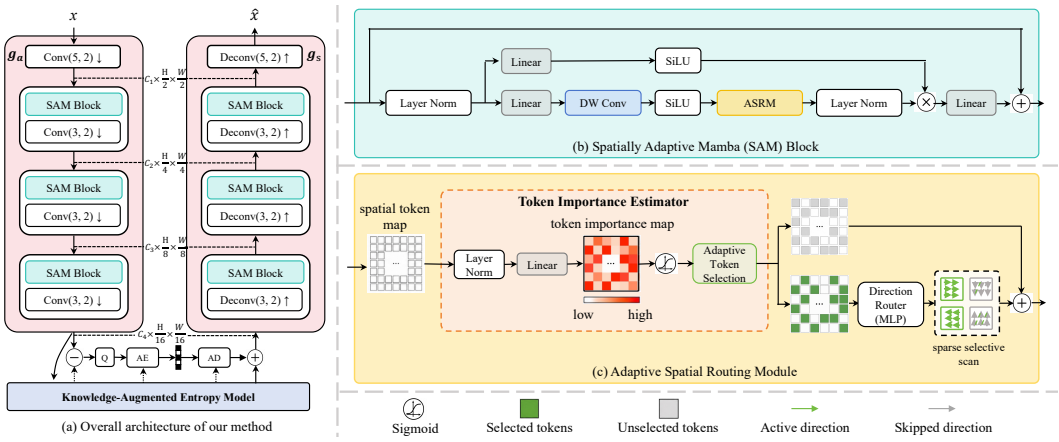


Figure 2: Overview of ASRIC. (a) SAM transforms are paired with KAEM; Q, AE, and AD denote quantization, arithmetic encoding, and arithmetic decoding. (b) Spatial routing is applied to the SAM content branch. (c) ASRM selects tokens and directions while retaining coordinates; gray directions are not executed.

$\hat{\mathbf{z}}$; $\hat{\mathbf{x}}$ is the reconstruction. The transform widths are $(128, 192, 256, 320)$. The primary latent has spatial size $H_y \times W_y$ and is partitioned into $G = 4$ contiguous groups of $C_g = 80$ channels.

Let $i \in \{1, \ldots, N\}$ index spatial positions in raster order, with $N = H_y W_y$, and let c index channels within a group. We use the factorization:

$$p(\hat{\mathbf{y}}, \hat{\mathbf{z}}) = p_{\hat{\mathbf{z}}}(\hat{\mathbf{z}}) \prod_{k=1}^{G} \prod_{i=1}^{N} \prod_{c=1}^{C_g} p\left(\hat{y}_{i,c}^{(k)} \mid \hat{\mathbf{z}}, \hat{\mathbf{y}}_{<i}^{(k)}, \hat{\mathbf{y}}^{<k}\right). \tag{2}$$

Here $\hat{\mathbf{y}}_{<i}^{(k)}$ contains preceding spatial positions in group k, and $\hat{\mathbf{y}}^{<k}$ contains all preceding groups. At a fixed (k, i), the channels are conditionally independent under the model. They share one prototype retrieval but have distinct probability parameters. The coding order is therefore group, spatial position, then channel; entropy routing never uses symbols at the current position in the current group. The two stages thus allocate resources without sharing routing decisions.

3.2 Spatially Adaptive Mamba

State-space models provide efficient long-range modeling, but conventional dense scanning applies the same recurrent computation to all spatial locations. SAM makes the number and locations of these updates content dependent while preserving dense local representations. Selection is learned through the RD objective and average compute constraints.

For a feature map $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, layer normalization produces $\mathbf{U} = \mathrm{LN}(\mathbf{X})$. Following the selective state-space formulation of Mamba (Gu & Dao, 2024), SAM adopts a gated local branch and a sparse long-range branch:

$$\begin{aligned} \mathbf{G} = \mathrm{SiLU}(\mathbf{W}_g \mathbf{U}), \qquad \mathbf{V} = \mathrm{SiLU}(\mathrm{DWConv}_{3\times3}(\mathbf{W}_v \mathbf{U})), \\ \mathrm{SAM}(\mathbf{X}) = \mathbf{X} + \mathbf{W}_o[\mathbf{G} \odot \mathrm{LN}(\mathrm{ASRM}(\mathbf{V}))], \end{aligned} \tag{3}$$

where $\mathbf{W}_g$, $\mathbf{W}_v$, and $\mathbf{W}_o$ are learnable projections, DWConv is depthwise convolution, SiLU is the sigmoid linear unit, and $\odot$ denotes elementwise multiplication. The selective state-space parameterization and discretization details are provided in Appendix A.1.

Adaptive spatial routing. ASRM predicts a content-dependent retention score for each token:

$$p_n = \sigma((\mathbf{w}_t^\top \mathrm{LN}(\mathbf{v}_n) + b_t)/T_t), \quad m_n^{\mathrm{h}} = \mathbb{I}[p_n \geq \tau_t], \quad \tilde{m}_n = \mathrm{stopgrad}(m_n^{\mathrm{h}} - p_n) + p_n, \tag{4}$$

where $\mathbf{v}_n$ denotes the feature token at spatial position n, $\mathbf{w}_t$ and b_t are learnable parameters of the token scorer, σ is the sigmoid, $\mathbb{I}[\cdot]$ is the indicator function, and stopgrad blocks gradient propagation. The temperature T_t controls score smoothness, and τ_t is the hard selection threshold. The

binary mask m_n^{h} is used during inference, while $\tilde{m}_n$ provides the straight-through gradient during training. A minimum count $K_{\min}$ is enforced by retaining the highest-scoring tokens when necessary. The budget losses constrain minibatch averages, so per-image counts can vary. No per-token marginal RD target is provided: a high score indicates a learned execution decision, not a measured local coding gain. Section 4.3 compares this policy with random, uniform-grid, feature-magnitude, and fixed-cardinality selection.

ASRM also selects propagation directions:

$$\mathbf{q} = \frac{\sum_n \tilde{m}_n \mathbf{v}_n}{\epsilon + \sum_n \tilde{m}_n}, \qquad \boldsymbol{\pi} = \sigma(f_{\mathrm{dir}}(\mathbf{q})/T_d), \qquad r_d^{\mathrm{h}} = \mathbb{I}[\pi_d \geq \tau_d], \tag{5}$$

where $\mathbf{q}$ is a masked pooled feature, $\epsilon > 0$ stabilizes the denominator, and f_{dir} is a learned direction scorer. For direction d, π_d is its sigmoid score, r_d^{h} is its binary execution decision, and T_d and τ_d are the temperature and threshold. An empty direction set is replaced by the highest-scoring direction.

Selected tokens retain their original coordinates after directional reorder. The recurrence scales its transition interval by the L1 displacement between successive retained coordinates. This geometric interval is distinct from the number of skipped scan positions, particularly at row or column boundaries (Appendix A.1). Appendix A.2 specifies the traversal orders and exact sparse execution procedure. Together, token and direction selection allocate recurrent updates while the dense branch retains local support at every location.

3.3 Knowledge-Augmented Entropy Model

The transform route allocates computation, whereas KAEM allocates statistical capacity. Edges, textures, and smooth regions repeatedly induce different conditional distributions, but a single dense predictor must represent all of them at once. KAEM stores dataset-level prototypes in specialized KBs and lets the current causal context select only the relevant banks. The memory is not a replacement for the hyperprior: it supplies reusable statistics that are fused with image-specific evidence.

Causal contexts. For channel group k and spatial position i, we form the causal context:

$$\begin{aligned} \boldsymbol{\Psi} &= h_s(\hat{\mathbf{z}}), & \boldsymbol{\Phi}_{\mathrm{sp},i}^{(k)} &= g_{\mathrm{sp}}(\hat{\mathbf{y}}_{<i}^{(k)})_i, \\ \boldsymbol{\Phi}_{\mathrm{ch},i}^{(k)} &= g_{\mathrm{ch}}(\hat{\mathbf{y}}^{<k})_i, & \mathbf{c}_i^{(k)} &= [\boldsymbol{\Psi}_i \| \boldsymbol{\Phi}_{\mathrm{sp},i}^{(k)} \| \boldsymbol{\Phi}_{\mathrm{ch},i}^{(k)}], \end{aligned} \tag{6}$$

where h_s is the hyper-synthesis transform, g_{sp} is a masked 5×5 spatial convolution, g_{ch} is a 1×1 projection of preceding groups, and $\|$ denotes channel concatenation. Their outputs $\boldsymbol{\Psi}_i$, $\boldsymbol{\Phi}_{\mathrm{sp},i}^{(k)}$, and $\boldsymbol{\Phi}_{\mathrm{ch},i}^{(k)}$ form the causal context $\mathbf{c}_i^{(k)}$. The spatial mask excludes the current and all later raster positions; the first group receives zero channel context. The hyperprior, spatial, and channel branches have widths 128, 128, and 64. Consequently, $\mathbf{c}_i^{(k)}$ is available before any channel at (k, i) is decoded.

Sparse knowledge retrieval. The memory contains N_{KB} KBs, each with M prototype values $\{\mathbf{m}_{n,m} \in \mathbb{R}^{d_m}\}_{m=1}^{M}$ and a routing key $\mathbf{u}_n \in \mathbb{R}^{d_r}$. Here n indexes banks, m indexes their prototypes, and d_m and d_r are the prototype and key dimensions. A readout projection maps prototype values to a 192-dimensional predictor embedding. A normalized query selects the Top-K keys, with K denoting the active-bank budget. Every selected bank performs dense attention over all M prototypes. Thus, KM prototypes are read per query from $N_{\mathrm{KB}}M$ stored entries: sparsity is at the bank level.

Relative to DCAE's dictionary retrieval, the mechanism adds a context-dependent bank support and a shared readout for each channel group. It does not introduce sparsity within a selected bank or establish that bank routing alone explains the complete-codec RD gain.

Keys and prototypes are initialized from a detached reservoir of training contexts collected during a deterministic warm-up. Keys use spherical clustering and values use within-cluster sampling. Prototype values then follow exponential moving average (EMA) updates, while routing and prediction parameters use gradient descent (Appendix A.3). Memory is frozen during coding.

For clarity, we suppress the group superscript below. Each i denotes a spatial position, and all retrieval quantities are shared across its C_g channels. Let $\mathbf{q}_i$ be the normalized 64-dimensional query derived from $\mathbf{c}_i$, and let $a_{i,n} = \mathbf{q}_i^\top \mathbf{u}_n / T_r$, where T_r is the bank-routing temperature. With $\mathcal{T}_i = \mathrm{TopK}(\mathbf{a}_i)$ denoting the indices of the K largest bank logits, the normalized bank weights are:

$$\omega_{i,n} = \frac{\mathbb{I}[n \in \mathcal{T}_i] e^{a_{i,n}}}{\sum_{j \in \mathcal{T}_i} e^{a_{i,j}}}. \tag{7}$$

The hard support is used in the forward pass and a dense-softmax surrogate is used for gradients. Within an activated bank, a second attention stage reads the prototype values:

$$\alpha_{i,n,m} = \mathrm{softmax}_m\left(\frac{(\mathbf{W}_k \mathbf{q}_i)^\top \mathbf{m}_{n,m}}{\sqrt{d_m}}\right),$$
$$\mathbf{k}_i = \mathbf{W}_v \left[\sum_{n \in \mathcal{T}_i} \omega_{i,n} \sum_{m=1}^{M} \alpha_{i,n,m} \mathbf{m}_{n,m}\right]. \tag{8}$$

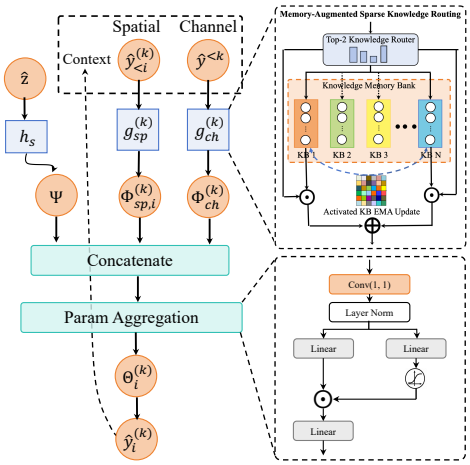


Figure 3: KAEM retrieves once per spatial position in each channel group. The shared readout supports channel-specific likelihoods without transmitting routing indices.

Here $\mathbf{W}_k$ and $\mathbf{W}_v$ are learned query and readout projections, $\alpha_{i,n,m}$ is the attention weight of prototype m in bank n, and $\mathbf{k}_i$ is the retrieved embedding. The operator $\mathrm{softmax}_m$ normalizes over prototypes within a bank. Equation 8 is evaluated once per (k, i) and its readout is reused for all C_g channels. The implementation tiles the selected-bank matrix products; unselected banks are not fetched. The linear readout projection is applied after aggregating the d_m-dimensional prototype values.

Probability parameterization. A pointwise convolution ($\mathrm{Conv}_{1\times1}$) fuses the readout with the causal context as $\mathbf{e}_i = \mathrm{LN}(\mathrm{Conv}_{1\times1}[\mathbf{c}_i \| \mathbf{k}_i])$. A gated projection then predicts the distribution parameters:

$$\mathbf{\Theta}_i = \mathbf{W}_o[(\mathbf{W}_a \mathbf{e}_i + \mathbf{b}_a) \odot \sigma(\mathbf{W}_g \mathbf{e}_i + \mathbf{b}_g)] + \mathbf{b}_o. \tag{9}$$

The projection matrices and biases are learned. The output $\mathbf{\Theta}_i \in \mathbb{R}^{C_g \times 3L}$ contains L Gaussian mixture components for each channel. Its c-th row specifies logits, means, and unconstrained scales $(\ell_{i,c,l}, \mu_{i,c,l}, \rho_{i,c,l})_{l=1}^{L}$. With $\beta_{i,c,l} = \mathrm{softmax}_l(\ell_{i,c,l})$ and $s_{i,c,l} = \mathrm{softplus}(\rho_{i,c,l}) + 10^{-4}$, which ensures positive scales, the scalar cumulative distribution function (CDF) and discretized mass are:

$$F_{i,c}(t) = \sum_{l=1}^{L} \beta_{i,c,l} \Phi\left(\frac{t - \mu_{i,c,l}}{s_{i,c,l}}\right),$$
$$p(\hat{y}_{i,c} \mid \mathbf{c}_i) = F_{i,c}(\hat{y}_{i,c} + \tfrac{1}{2}) - F_{i,c}(\hat{y}_{i,c} - \tfrac{1}{2}). \tag{10}$$

Here Φ is the standard normal CDF and $\mathrm{softplus}(x) = \log(1 + e^x)$. Sharing a retrieved feature does not tie mixture parameters across channels. A factorized learned CDF codes the hyperlatent; identical float32 evaluation and 16-bit CDF integerization are used at both codec ends. The causal schedule and retrieval cost are given in Appendix A.4. Adaptive bank selection therefore changes predictive statistics while preserving the causal schedule and retrieval granularity. Appendix A.10 reports lower residual spatial correlations for ASRIC in standardized latents $(\hat{y} - \mu)/\sigma$.

3.4 Training Objective and Coding Consistency

Joint RD training coordinates the routes, with separate constraints on computation and bank utilization. Uniform noise approximates quantization during training and rounding is used for coding. Let $p_{\hat{\mathbf{y}}}$ and $p_{\hat{\mathbf{z}}}$ be the joint masses assigned to the primary and hyper latents. The objective is:

$$\mathcal{L} = \underbrace{\mathbb{E}[-\log_2 p_{\hat{\mathbf{y}}} - \log_2 p_{\hat{\mathbf{z}}}] + \lambda d(\mathbf{x}, \hat{\mathbf{x}})}_{\mathcal{L}_{\mathrm{RD}}} + \eta_t \mathcal{L}_{\mathrm{tok}} + \eta_d \mathcal{L}_{\mathrm{dir}} + \eta_b \mathcal{L}_{\mathrm{bal}}. \tag{11}$$

Here d is reconstruction distortion, λ controls the RD trade-off, and η_t, η_d, and η_b weight the routing losses. The token and direction losses $\mathcal{L}_{\mathrm{tok}}$ and $\mathcal{L}_{\mathrm{dir}}$ constrain minibatch-average budgets; $\mathcal{L}_{\mathrm{bal}}$ balances bank usage. Hard decisions are used in the forward pass with straight-through gradients. At inference, only packed selected tokens are scanned, and inactive directions are not launched. Since KAEM uses only decoder-causal context and frozen memory, both sides can reproduce the same Top-K selection and channel-specific CDFs under identical numerical evaluation. Analysis and synthesis masks are computed locally from each transform's inputs; neither requires a transmitted mask. Adaptive allocation is therefore compatible with independent routing and deterministic decoding.

4 Experiments

4.1 Implementation and Evaluation Protocol

Training. ASRIC is trained on a fixed 500,000-image subset of ImageNet-1k (Deng et al., 2009), with random 256×256 crops, flips, batch size 16, and 300 epochs at $\lambda \in \{0.0035, 0.0067, 0.0130, 0.0250, 0.0500\}$. The four transform widths are $\{C_1, C_2, C_3, C_4\} = \{128, 192, 256, 320\}$ and the C_4 latent is split into four 80-channel groups. The default MASKR uses $N_{\mathrm{KB}} = 8$ banks with $M = 16{,}384$ prototypes each, key dimension $d_r = 64$, value dimension $d_m = 88$, and an active-bank budget of $K = 2$. Each channel is modeled by $L = 5$ Gaussian mixture components. Adam, initialization, and coder details are in Appendix A.5.

Evaluation and reference codec. Kodak (24), CLIC Professional Validation (41), and Tecnick (100) are coded at native resolution (Eastman Kodak Company, 1993; Toderici et al., 2020; Asuni & Giachetti, 2014). The hyperlatent is coded first, followed by the four primary-latent groups in the order defined in Section 3.1; rate includes both bitstreams and headers. BD-Rate uses the same five-point peak signal-to-noise ratio (PSNR) integration against the VVC Test Model 9.1, denoted VTM-9.1 throughout (Bross et al., 2021; Bjøntegaard, 2001). VTM-9.1 uses its official intra configuration as the reference curve.

Baseline training and evaluation. ASRIC and all learned baselines are trained on the same dataset. Baselines use their official implementations where available, with weights trained on the common data. A shared evaluation wrapper standardizes hardware, software, input preparation, bit accounting, and timing. Appendix A.6 specifies the measurement protocol.

Complexity. Floating-point operations (FLOPs) and time are per-image CLIC averages at native resolution, batch size one. FLOPs denote the sum of encoder and decoder neural operations. Prototype attention is counted once per spatial position in each channel group, with channel-specific prediction counted separately (Appendix A.4). Learned routing projections count toward FLOPs; index selection, gather, reorder, packing, and scatter are charged to latency. Time also includes arithmetic coding and excludes file I/O. Appendix A.7 compares RTX 4090 and A100 timings.

4.2 RD and Complexity Comparison

Table 1 compares convolutional neural network (CNN), Transformer, SSM, hierarchical-context, dictionary, and graph-based codecs (Cheng et al., 2020; Liu et al., 2023; Li et al., 2024; 2025; Lu et al., 2025; Chen et al., 2026a; Zeng et al., 2025; Chen et al., 2026b). ASRIC achieves BD-Rate reductions of 19.15%, 18.67%, and 20.37% relative to VTM-9.1 on Kodak, CLIC, and Tecnick. CMIC is the strongest learned baseline in this comparison and a closely related content-adaptive SSM codec. The improvements over CMIC are 0.94, 0.81, and 0.32 percentage points in VTM-referenced BD-Rate, respectively. These are score differences, not direct pairwise BD-Rates against CMIC. ASRIC uses 2.29 versus 2.39 TFLOPs and takes 0.67/0.41 s to encode/decode, compared with CMIC's 0.71/0.49 s. It therefore improves RD with 4.2% fewer nominal FLOPs, 5.6% shorter encoding time, and 16.3% shorter decoding time.

DCAE is the closest dictionary-based comparison. ASRIC improves its Kodak, CLIC, and Tecnick scores by 1.43, 1.18, and 1.44 points, respectively, with 2.29 versus 2.36 TFLOPs. Decoding is faster (0.41 versus 0.49 s), whereas encoding is slower (0.67 versus 0.51 s). Relative to MambaIC,

Table 1: RD and complexity comparison. BD-Rate is reported relative to VTM-9.1. FLOPs denote the total encoder–decoder neural operations, and runtime is measured as CLIC per-image averages on a single RTX 4090.

Method	Complexity			BD-Rate (%)		
	FLOPs (T)	Enc. (s)	Dec. (s)	Kodak	CLIC	Tecnick
VTM-9.1	–	–	–	0.00	0.00	0.00
Cheng2020-attn (Cheng et al., 2020)	2.03	5.31	>10	+4.46	+5.92	+3.97
TCM-L (Liu et al., 2023)	3.74	0.54	0.49	-13.47	-9.16	-11.01
FTIC (Li et al., 2024)	2.38	>10	>10	-14.69	-10.54	-14.27
HPCM (Li et al., 2025)	2.00	0.46	0.45	-16.29	-15.18	-17.42
DCAE (Lu et al., 2025)	2.36	0.51	0.49	-17.72	-17.49	-18.93
GLIC (Chen et al., 2026a)	2.48	0.53	0.36	-18.14	-17.53	-19.17
MambaIC (Zeng et al., 2025)	5.56	1.17	0.58	-15.28	-15.87	-15.91
CMIC (Chen et al., 2026b)	2.39	0.71	0.49	-18.21	-17.86	-20.05
ASRIC (Ours)	2.29	0.67	0.41	**-19.15**	**-18.67**	**-20.37**

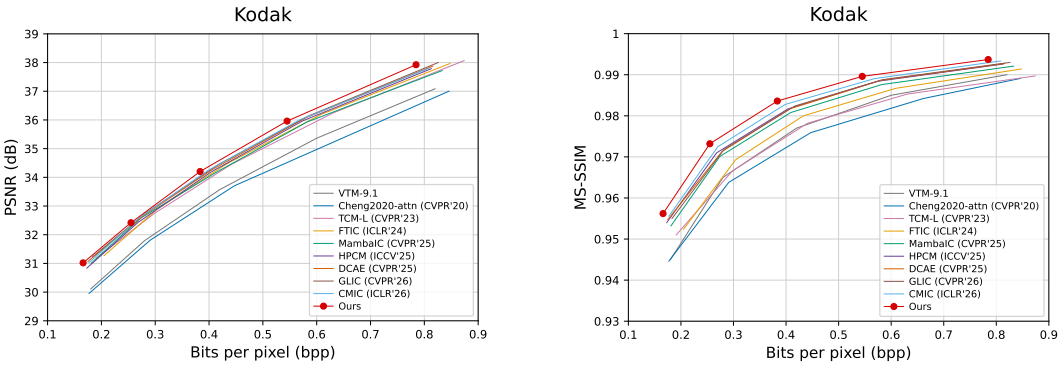


Figure 4: RD performance on Kodak: PSNR (left) and MS-SSIM (right).

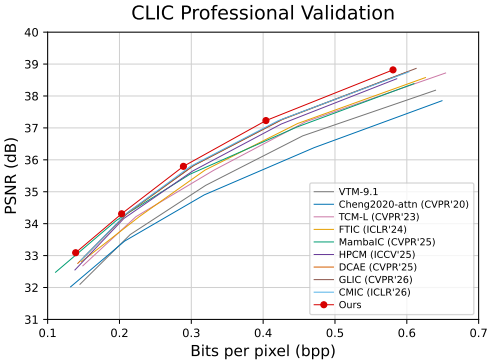


Figure 5: PSNR–rate curves on CLIC.

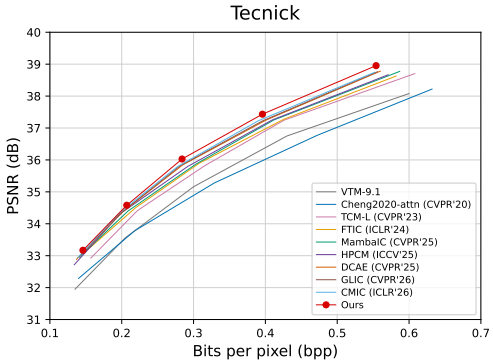


Figure 6: PSNR–rate curves on Tecnick.

nominal FLOPs fall by 58.8% and decoding time by 29.3% (0.58 to 0.41 s). These are complete-codec comparisons; they do not isolate the effect of bank routing from differences in transforms and entropy predictors.

Figures 4–6 show the RD curves over the evaluated bitrate range. Figure 4 reports both PSNR and multiscale structural similarity (MS-SSIM) on Kodak; Figures 5 and 6 report PSNR on CLIC and Tecnick.

4.3 Ablation Study and Analysis

We analyze ASRM routing, its interaction with MASKR, and sparse scanning. The default configuration is selected on ImageNet validation data and fixed before test-set evaluation. Kodak ablations provide post hoc diagnostics and do not determine the defaults. Unless otherwise stated, variants share the network configuration, coding order, and training budget apart from the factor under study. Appendices A.8 and A.9 report capacity sensitivity and additional diagnostic controls.

ASRM controls. Table 2 compares spatial selection policies at similar nominal cost. Dynamic TIE improves the Kodak BD-Rate score by 1.47 percentage points over random selection, 1.13 over a uniform grid, and 0.68 over feature-magnitude ranking. It also improves on fixed Top-K_t TIE by 0.32 points, where K_t denotes the retained-token count. The controls span 2.27–2.34 TFLOPs, placing these comparisons within a narrow range of nominal cost. With dynamic TIE fixed, replacing unit intervals with L1 geometric intervals improves the score from -18.76% to -19.15%, at an additional 0.02 TFLOPs, supporting coordinate-dependent transitions within the sparse configuration.

Table 2: ASRM controls. Kodak denotes PSNR BD-Rate (%) versus VTM-9.1; FLOPs are CLIC averages. L1 denotes geometric intervals.

Selector	Gap rule	Kodak	FLOPs (T)
Random	L1	-17.68	2.31
Uniform grid	L1	-18.02	2.27
Feature magnitude	L1	-18.47	2.34
Fixed Top-K_t	L1	-18.83	2.30
Dynamic TIE	unit	-18.76	2.27
Dynamic TIE	L1	**-19.15**	2.29

Joint routing. Table 3 shows that both routes improve RD performance, with their combination achieving the best scores on all three datasets. Adding ASRM to MASKR improves the Kodak, CLIC, and Tecnick scores by 1.13, 1.05, and 1.16 percentage points, respectively. With the same eight-bank configuration, dense retrieval achieves -19.20% on Kodak at 2.73 TFLOPs, compared with -19.15% at 2.29 TFLOPs for Top-2 (Appendix A.9). Reading all banks improves the score by only 0.05 points while adding 0.44 TFLOPs. This comparison quantifies the RD cost of restricting active retrieval within ASRIC; the comparison with DCAE also includes differences in the transform and entropy predictor.

Table 3: Module ablation. PSNR BD-Rate (%) is relative to VTM-9.1.

ASRM	MASKR	Kodak	CLIC	Tecnick
		-15.26	-14.72	-16.58
✓		-16.31	-15.54	-17.14
	✓	-18.02	-17.62	-19.21
✓	✓	**-19.15**	**-18.67**	**-20.37**

Dynamic sparse scanning. Appendix A.2 varies the cap on active directions. The default uses one to four directions, averaging about two. Relative to full scanning, FLOPs fall from 5.31 to 2.29 T (56.9%), while decoding falls from 0.48 to 0.41 s (14.6%, or $1.17\times$ speedup), at a 0.17-point Kodak BD-Rate penalty (-19.32% to -19.15%). Dense local operations, entropy inference, and arithmetic coding remain, while routing and data movement add overhead. FLOP savings thus need not translate proportionally into latency savings; attributing the residual cost requires operator profiling.

Qualitative results. Appendix A.12 presents Kodak reconstructions and local detail comparisons alongside their bitrates and PSNR values.

5 Conclusion

ASRIC adapts transform computation and entropy-model capacity through coordinate-dependent sparse scans and decoder-reproducible Top-K bank retrieval. The two routes provide complementary RD gains on Kodak, CLIC, and Tecnick without transmitting routing indices. Relative to full scanning, ASRIC reduces nominal FLOPs by 56.9% and decoding time by 14.6%, with a 0.17-point Kodak BD-Rate penalty. These results quantify the RD–computation trade-off and distinguish arithmetic savings from end-to-end acceleration.

Reproducibility Statement

Sections 3.1–3.4 define the codec, shared retrieval, and scalar likelihoods. Appendices A.1–A.4 specify memory updates and sparse execution. Training, coding, and evaluation protocols are given in Section 4.1 and Appendices A.5–A.6.

AI Use Statement

Artificial intelligence assistance was used solely to refine and polish the English phrasing, language style, and grammar of this manuscript. All conceptual formulations, methodology design, theoretical derivations, experimental evaluations, and analytical conclusions are entirely the work of the authors.

References

Nicola Asuni and Andrea Giachetti. Testimages: A large-scale archive for testing visual devices and basic image processing algorithms. In *Smart Tools and Apps for Graphics*, pp. 63–70, 2014.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkcQFMZRb.

Gisle Bjøntegaard. Calculation of average PSNR differences between RD-curves. Technical Report VCEG-M33, ITU-T Video Coding Experts Group, 2001.

Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3736–3764, 2021.

Yunuo Chen, Bing He, Zezheng Lyu, Hongwei Hu, Qunshan Gu, Yuan Tian, and Guo Lu. Adaptive learned image compression with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12150–12161, 2026a.

Yunuo Chen, Zezheng Lyu, Bing He, Hongwei Hu, Qi Wang, Yuan Tian, Li Song, Wenjun Zhang, and Guo Lu. Content-aware mamba for learned image compression. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=WwDNiisZQm.

Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7939–7948, 2020.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10041–10071. PMLR, 2024. URL https://proceedings.mlr.press/v235/dao24a.html.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Eastman Kodak Company. Kodak lossless true color image suite. http://r0k.us/graphics/kodak/, 1993.

Donghui Feng, Zhengxue Cheng, Shen Wang, Ronghua Wu, Hongwei Hu, Guo Lu, and Li Song. Linear attention modeling for learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7623–7632, 2025.

Haisheng Fu, Jie Liang, Zhenman Fang, Jingning Han, Feng Liang, and Guohe Zhang. WeConvene: Learned image compression with wavelet-domain convolution and entropy model. In *European Conference on Computer Vision*, pp. 37–53, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Hang Guo, Yong Guo, Yaohua Zha, Yulun Zhang, Wenbo Li, Tao Dai, Shu-Tao Xia, and Yawei Li. MambaIRv2: Attentive state space restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28124–28133, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Guo_MambaIRv2_Attentive_State_Space_Restoration_CVPR_2025_paper.html.

Minghao Han, Shiyin Jiang, Shengxi Li, Xin Deng, Mai Xu, Ce Zhu, and Shuhang Gu. Causal context adjustment loss for learned image compression. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14771–14780, 2021.

Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. ELIC: Efficient learned image compression with unevenly grouped space–channel contextual adaptive coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5718–5727, 2022.

Wei Jiang, Jiayu Yang, Yongqi Zhai, Peirong Ning, Feng Gao, and Ronggang Wang. MLIC: Multi-reference entropy model for learned image compression. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 7618–7627, 2023.

Jun-Hyuk Kim, Byeongho Heo, and Jong-Seok Lee. Joint global and local hierarchical priors for learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5992–6001, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Kim_Joint_Global_and_Local_Hierarchical_Priors_for_Learned_Image_Compression_CVPR_2022_paper.html.

Han Li, Shaohui Li, Wenrui Dai, Chenglin Li, Junni Zou, and Hongkai Xiong. Frequency-aware transformer for learned image compression. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=HKGQDDTuvZ.

Yuqi Li, Haotian Zhang, Li Li, and Dong Liu. Learned image compression with hierarchical progressive context modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18834–18843, 2025.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=BjyvwnXXVn_.

Jinming Liu, Heming Sun, and Jiro Katto. Learned image compression with mixed transformer–CNN architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14388–14397, 2023.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: Visual state space model. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Jingbo Lu, Leheng Zhang, Xingyu Zhou, Mu Li, Wen Li, and Shuhang Gu. Learned image compression with dictionary-based entropy model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12850–12859, 2025.

David Minnen, Johannes Ballé, and George D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Yichen Qian, Ming Lin, Xiuyu Sun, Zhiyu Tan, and Rong Jin. Entroformer: A transformer-based entropy model for learned image compression. In *International Conference on Learning Representations*, 2022.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8583–8595, 2021.

George Toderici, Wenzhe Shi, Radu Timofte, Lucas Theis, Johannes Ballé, Eirikur Agustsson, Nick Johnston, and Fabian Mentzer. CLIC image compression benchmark. `https://www.compression.cc/`, 2020.

Fei Xie, Weijia Zhang, Zhongdao Wang, and Chao Ma. QuadMamba: Learning quadtree-based selective scan for visual state space model. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Yibo Yang and Stephan Mandt. Computationally-efficient neural image compression with shallow decoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 530–540, 2023. URL `https://openaccess.thecvf.com/content/ICCV2023/html/Yang_Computationally-Efficient_Neural_Image_Compression_with_Shallow_Decoders_ICCV_2023_paper.html`.

Hongxu Yin, Arash Vahdat, Jose M. Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10809–10818, 2022.

Fanhu Zeng, Hao Tang, Yihua Shao, Siyu Chen, Ling Shao, and Yan Wang. MambaIC: State space models for high-performance learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18041–18050, 2025.

Zheng Zhan, Zhenglun Kong, Yifan Gong, Yushu Wu, Zichong Meng, Hangyu Zheng, Xuan Shen, Stratis Ioannidis, Wei Niu, Pu Zhao, and Yanzhi Wang. Exploring token pruning in vision state space models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M. Dai, Zhifeng Chen, Quoc V. Le, and James Laudon. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*, volume 35, pp. 7103–7114, 2022.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62429–62442. PMLR, 2024. URL `https://proceedings.mlr.press/v235/zhu24f.html`.

A Appendix

A.1 Selective State-space Parameterization

This section provides additional implementation details of the selective state-space model used in SAM. Following the selective state-space formulation of Mamba (Gu & Dao, 2024), the continuous-time dynamics are defined as:

$$\frac{d\mathbf{h}(t)}{dt} = \mathbf{A}\mathbf{h}(t) + \mathbf{B}(t)\mathbf{x}(t), \qquad \mathbf{y}(t) = \mathbf{C}(t)\mathbf{h}(t) + \mathbf{D}\mathbf{x}(t), \tag{12}$$

where $\mathbf{h}(t)$ denotes the hidden state, $\mathbf{x}(t)$ and $\mathbf{y}(t)$ denote the input and output features, $\mathbf{A}$ is the state transition matrix, $\mathbf{B}(t)$ and $\mathbf{C}(t)$ are input-dependent projection matrices, and $\mathbf{D}$ is a learned skip parameter.

To ensure stable state evolution, the transition matrix is parameterized as:

$$\mathbf{A} = -\exp(\mathbf{A}_{\log}), \tag{13}$$

where $\mathbf{A}_{\log}$ is a learnable diagonal parameter.

For each input token $\mathbf{x}_j$, the selective parameters are generated by linear projections:

$$\mathbf{B}_j = W_B\mathbf{x}_j, \qquad \mathbf{C}_j = W_C\mathbf{x}_j, \tag{14}$$

and the input-dependent discretization interval is computed as:

$$\Delta_j = \mathrm{softplus}(\mathbf{w}_\Delta^\top \mathbf{x}_j + b_\Delta) + 10^{-4}, \tag{15}$$

where W_B, W_C, $\mathbf{w}_\Delta$, and b_Δ are learnable parameters, and Δ_j controls the state evolution scale of token j. The state size is set to $S = 16$.

Using zero-order hold discretization, the continuous system is converted into the recurrent form:

$$\bar{\mathbf{A}}_j = \exp(\Delta_j\mathbf{A}), \qquad \bar{\mathbf{B}}_j = \left(\int_0^{\Delta_j} \exp(s\mathbf{A})\, ds\right)\mathbf{B}_j. \tag{16}$$

The recurrent update is expressed as:

$$\mathbf{h}_j = \bar{\mathbf{A}}_j\mathbf{h}_{j-1} + \bar{\mathbf{B}}_j\mathbf{x}_j, \qquad \mathbf{o}_j = \mathbf{C}_j\mathbf{h}_j + \mathbf{D}\mathbf{x}_j. \tag{17}$$

In SAM, the selective transition is independently applied along four scan directions $\mathcal{D} = \{\rightarrow, \leftarrow, \downarrow, \uparrow\}$. All directions share the same state-space parameters while using different token traversal orders.

Coordinate-preserving sparse transition. ASRM uses a geometric interval on the retained coordinates. The selective parameterization is unchanged, but the recurrence is evaluated only at retained tokens. For consecutive retained coordinates, define:

$$g_{d,1} = 1, \qquad g_{d,j} = \|\mathbf{p}_{d,j} - \mathbf{p}_{d,j-1}\|_1, \qquad \delta_{d,j} = g_{d,j}\left[\mathrm{softplus}(\mathbf{w}_\delta^\top \mathbf{x}_{d,j} + b_\delta) + 10^{-4}\right], \tag{18}$$

where $\mathbf{p}_{d,j}$ and $\mathbf{x}_{d,j}$ are the coordinate and feature of the j-th retained token along direction d. The learnable step projection $\mathbf{w}_\delta$ and bias b_δ determine its content-dependent interval. The term $g_{d,j}$ is the Manhattan displacement on the feature grid. It is neither the compacted index difference nor, in general, the distance along the original scan path. To distinguish these notions, let κ_d be the uncompacted directional rank defined in Eq. 21, and set:

$$s_{d,j} = \kappa_d(\mathbf{p}_{d,j}) - \kappa_d(\mathbf{p}_{d,j-1}), \qquad n_{d,j}^{\mathrm{skip}} = s_{d,j} - 1 \quad (j > 1). \tag{19}$$

Here $s_{d,j}$ counts scan transitions, and $n_{d,j}^{\mathrm{skip}}$ counts omitted positions. Neither quantity replaces $g_{d,j}$ in the reported model.

For a width-eight row-major scan, $(0,1) \to (0,4)$ has $g = s = 3$ and skips two positions. At a row boundary, $(0,7) \to (1,0)$ has $g = 8$ but $s = 1$: no position is skipped. Conversely, $(0,1) \to (1,1)$ has $g = 1$ and $s = 8$, with seven positions skipped. Column-major scans have the corresponding

column-boundary case. Thus, geometric distance can either exceed or fall below scan-rank distance, and even a dense raster traversal can have $g > 1$ at a wrap.

The corresponding coordinate-aware state transition is formulated as:

$$\begin{aligned} \mathbf{h}_{d,j} &= \exp(\delta_{d,j}\mathbf{A})\mathbf{h}_{d,j-1} + \left(\int_0^{\delta_{d,j}} \exp(s\mathbf{A})\, ds\right) \mathbf{B}_{d,j}\mathbf{x}_{d,j}, \\ \mathbf{o}_{d,j} &= \mathbf{C}_{d,j}\mathbf{h}_{d,j} + \mathbf{D}\mathbf{x}_{d,j}. \end{aligned} \tag{20}$$

When retained coordinates are adjacent on the grid, $g_{d,j} = 1$ recovers the unit geometric interval. Larger L1 displacement increases the transition interval and therefore attenuates the carried state more strongly because $\mathbf{A}$ is negative diagonal. This is a geometric inductive bias. It does not exactly integrate the skipped dense selective updates, whose input-dependent steps and input contributions have not been evaluated.

The reported rule carries the state across row and column boundaries and uses the same L1 interval there. Table 2 compares this rule only with unit intervals. It does not isolate boundary handling or establish superiority over scan-rank intervals. A controlled comparison must hold token masks and active directions fixed and vary L1 intervals, rank intervals, and state resets at line boundaries; each changes the model's inductive bias.

A.2 Sparse Scan Derivation and Exact Execution

For zero-based coordinate (r, c) on an $H \times W$ feature grid, the fixed ranks used to form directional sequences are:

$$\begin{aligned} \kappa_{\rightarrow}(r,c) &= rW + c, \quad \kappa_{\leftarrow}(r,c) = rW + (W-1-c), \\ \kappa_{\downarrow}(r,c) &= cH + r, \quad\;\; \kappa_{\uparrow}(r,c) = cH + (H-1-r). \end{aligned} \tag{21}$$

Because these ranks depend only on (H, W), direction-specific index lists can be precomputed once for each resolution. The hard selection mask is applied to each list, so a generic data-dependent comparison sort is not required. Ties in the token threshold are broken by the flattened row-major index. The first coordinate gap is one by definition; subsequent L1 displacements are computed after this fixed reorder, including row or column boundaries. Horizontal reversal reverses each row and vertical reversal reverses each column under these definitions; neither reverses the entire flattened grid.

Algorithm 1 Coordinate-preserving sparse selective scan (inference)

1: **Input:** feature map $\mathbf{V}$, coordinate grid $\mathcal{P}$, trained token and direction scorers.
2: Compute scores p_n and hard index set $\mathcal{I}$ using Eq. 4.
3: Insert the highest-scoring indices if $|\mathcal{I}| < K_{\min}$.
4: Gather $\mathbf{V}_{\mathcal{I}} = \mathrm{Gather}(\mathbf{V}, \mathcal{I})$ and $\mathcal{P}_{\mathcal{I}} = \mathcal{P}[\mathcal{I}]$.
5: Obtain active directions $\mathcal{D}_{\mathbf{V}}$ using Eq. 5; retain the highest-scoring direction if empty.
6: Initialize $\mathbf{Z}_{\mathcal{I}} \leftarrow \mathbf{0}$.
7: **for** $d \in \mathcal{D}_{\mathbf{V}}$ **do**
8: Reorder $(\mathbf{V}_{\mathcal{I}}, \mathcal{P}_{\mathcal{I}})$ using κ_d.
9: Compute L1 displacements and run Eq. 20; carry the state across line boundaries.
10: Inverse-reorder and accumulate the result into $\mathbf{Z}_{\mathcal{I}}$.
11: **end for**; normalize by $|\mathcal{D}_{\mathbf{V}}|$.
12: **Return:** the bypassed unselected map plus $\mathrm{Scatter}(\mathbf{Z}_{\mathcal{I}}, \mathcal{I})$.

Here $\mathcal{I}$ contains retained spatial indices, $\mathcal{D}_{\mathbf{V}}$ is the active direction set, and $\mathbf{Z}_{\mathcal{I}}$ accumulates retained-token outputs. Gather collects entries at $\mathcal{I}$; Scatter restores them to the feature grid with zeros elsewhere. Unselected features bypass the scan. The referenced routing equations include the temperatures, thresholds, and masked pooling used by the implementation.

For $K_{\mathbf{X}}$ retained tokens, $D_{\mathbf{X}}$ active directions, channel width C, and state size S, the selective recurrence requires $\mathcal{O}(K_{\mathbf{X}} D_{\mathbf{X}} CS)$ work. Coordinate gathering, packing, and scattering are linear in the

number of retained indices; filtering the fixed-rank lists requires a linear pass over the feature grid. These terms are included in the reported latency and are not removed from the practical complexity measurement. In the fixed-cardinality controls of Table 4, $k = q$ means a Top-q cap on the four direction logits; it is unrelated to the channel-group index in Eq. 2.

Table 4: Dynamic scan analysis. $k = q$ denotes a fixed Top-q cap on active directions; token selection remains dynamic. BD-Rate is on Kodak; FLOPs (total encoder+decoder) and decoding time are CLIC averages.

Scan policy	FLOPs (T)	Dec. (s)	BD-Rate (%)
Full scan	5.31	0.48	**-19.32**
Uniform sparse scan	2.61	0.39	-17.86
Dynamic scan ($k = 1$)	1.72	0.33	-18.31
Dynamic scan ($k = 2$)	2.38	0.38	-18.79
Dynamic scan ($k = 3$)	3.62	0.44	-19.05
ASRIC	2.29	0.41	-19.15

A.3 Sparse Knowledge Routing and Memory Updates

Let $\mathcal{B}$ contain the image–group–position triplets $u = (b, k, i)$ in a minibatch. Each shared retrieval contributes one assignment, irrespective of the number of channels at that position. Let $\mathbf{t}_u = \mathrm{sg}(\mathbf{W}_m \mathbf{c}_u) \in \mathbb{R}^{88}$ be the target prototype-value embedding, where $\mathbf{W}_m$ projects the causal context into the prototype-value space. Here sg denotes the same stop-gradient operator as $\mathrm{stopgrad}$ in Eq. 4: it is the identity in the forward pass and has zero derivative. The assignment mass of prototype (n, m) is:

$$\gamma_{u,n,m} = \mathbb{I}[n \in \mathcal{T}_u]\, \omega_{u,n} \alpha_{u,n,m}. \tag{22}$$

If $\sum_{u \in \mathcal{B}} \gamma_{u,n,m} > 0$, its EMA sufficient statistics and prototype are updated by:

$$C_{n,m} \leftarrow \mu C_{n,m} + (1 - \mu) \sum_{u \in \mathcal{B}} \gamma_{u,n,m}, \tag{23}$$

$$\mathbf{S}_{n,m} \leftarrow \mu \mathbf{S}_{n,m} + (1 - \mu) \sum_{u \in \mathcal{B}} \gamma_{u,n,m} \mathbf{t}_u, \tag{24}$$

$$\mathbf{m}_{n,m} \leftarrow \mathbf{S}_{n,m} / (C_{n,m} + \epsilon). \tag{25}$$

Here $C_{n,m}$ is the running assignment mass, $\mathbf{S}_{n,m}$ is the weighted embedding sum, μ is the EMA decay, and $\epsilon > 0$ stabilizes the denominator. Unassigned prototypes are left unchanged rather than decayed. Before these updates, the initialization described in Section 3.3 collects a detached reservoir of at most 131,072 context projections during a 2,000-minibatch warm-up with memory reads disabled, obtains eight spherical key centroids with k-means++ (seed 2024), and fills each 16,384-entry bank by sampling the corresponding 88-dimensional value projections. Sampling uses replacement only for under-populated banks; EMA counts start at one and sums at the sampled values. The reservoir and the centroids are discarded after initialization; the prototype values and keys are the only persistent memory used during coding.

For a batch of B feature maps, the average-compute constraints are:

$$\mathcal{L}_{\mathrm{tok}} = \left(\frac{1}{B} \sum_{b=1}^{B} \frac{1}{N^{(b)}} \sum_{n=1}^{N^{(b)}} p_{b,n} - \rho \right)^2, \tag{26}$$

$$\mathcal{L}_{\mathrm{dir}} = \left(\frac{1}{B} \sum_{b=1}^{B} \sum_{d \in \mathcal{D}} \pi_{b,d} - \kappa \right)^2, \tag{27}$$

where $N^{(b)}$ is the number of spatial tokens in image b at the current block. The scores $p_{b,n}$ and $\pi_{b,d}$ follow Eqs. 4 and 5; ρ and κ are the target mean keep ratio and direction count. These losses do not prescribe a per-image count.

For the knowledge router, write $\Omega = \mathcal{B}$ for the same triplet set, counted once per shared retrieval. Let $\pi_{u,n}$ denote the dense softmax probability over all KB keys before Top-K truncation. Load balancing is imposed through:

$$\mathcal{L}_{\text{bal}} = N_{\text{KB}} \sum_{n=1}^{N_{\text{KB}}} \bar{\pi}_n \bar{f}_n, \qquad \bar{\pi}_n = \frac{1}{|\Omega|} \sum_{u \in \Omega} \pi_{u,n}, \qquad \bar{f}_n = \frac{1}{K|\Omega|} \sum_{u \in \Omega} \mathbb{I}[n \in \mathcal{T}_u]. \tag{28}$$

For router learning, hard Top-K weights are used in the forward pass and $\tilde{\boldsymbol{\omega}} = \text{sg}(\boldsymbol{\omega} - \boldsymbol{\pi}) + \boldsymbol{\pi}$ is used in the backward pass. All four scan directions may be materialized during optimization to provide stable surrogate gradients; the inference implementation dispatches only the hard active set.

A.4 Spatially Shared Retrieval and Causal Coding

The entropy model uses one query for each spatial position within a channel group. For a latent grid of $H_y \times W_y$ and G groups, there are GH_yW_y retrievals per encoding or decoding pass. The readout at (k, i) is shared by the C_g channels, while the parameter head emits $3LC_g$ values and constructs a separate scalar CDF for each channel.

Decoding schedule. First decode $\hat{\mathbf{z}}$ and compute $\boldsymbol{\Psi}$. Process groups in increasing k and positions in raster order. At each (k, i), construct $\mathbf{c}_i^{(k)}$ from preceding groups and preceding spatial positions in the current group. Compute Top-K routing and prototype attention once, retain the readout $\mathbf{k}_i^{(k)}$, and predict all C_g sets of mixture parameters. Decode channels in increasing channel index using these CDFs. Only after the position is complete does its decoded channel vector enter the spatial context of subsequent positions. The encoder uses the same masks and ordering. Memory remains fixed throughout the coding pass.

This schedule is causal because the context at (k, i) excludes every current and future symbol in group k. Starting from the shared hyperlatent and model state, identical previously decoded symbols yield identical queries, bank indices, and CDFs when the same numerical kernels and tie-breaking rules are used. The shared readout changes the granularity of probability computation; each scalar symbol still has its own mass in Eq. 2.

Retrieval complexity. Let K denote the number of active banks and $d_e = 192$ the readout width. The prototype dot products and weighted aggregation require $\mathcal{O}(KMd_m)$ work per query. Since $\mathbf{W}_v$ is linear, it can be applied once to the aggregated d_m-dimensional vector, with cost $\mathcal{O}(d_m d_e)$. Including comparisons with the bank keys, the retrieval cost per coding pass is:

$$\mathcal{O}(GH_yW_y[N_{\text{KB}}d_r + KMd_m + d_m d_e]). \tag{29}$$

Thus, the prototype-attention term is charged once per group and position, not once per scalar channel. Channel-specific parameter prediction and CDF construction remain necessary. With a multiply and an addition counted as two operations, the dot products and value aggregation contribute approximately $4KMd_m$ FLOPs per query, excluding softmax, projections, and routing.

For example, a 768×512 image produces a 48×32 latent grid and $4 \times 48 \times 32 = 6{,}144$ queries per pass. With the default memory, the dot-product and aggregation terms amount to approximately 70.87 GFLOPs per pass. This analytic count describes only prototype attention; it is not a measurement of full-codec FLOPs or latency. Encoder and decoder retrievals are counted separately, and the measurement protocol also covers prediction, transforms, and the other dispatched neural operators.

A.5 Optimization and Reproducibility Details

The 500,000-image training subset is sampled once with seed 2024 from the ImageNet-1k training split; ImageNet validation images are used only for model selection. Images whose shorter side is below 256 pixels are first resized so that a valid crop can be drawn. Pixel values are represented in $[0, 1]$ without mean–variance normalization. The four transform widths are $(C_1, C_2, C_3, C_4) = (128, 192, 256, 320)$; the $C_4 = 320$ primary-latent channels are divided into four contiguous 80-channel groups. The learning rate is reduced from 10^{-4} to 2×10^{-5}, 5×10^{-6}, and 10^{-6} at epochs 240, 270, and 285, respectively. The global gradient norm is clipped at 1.0.

The token and direction thresholds are both 0.5. Their sigmoid temperatures are annealed linearly from 1.0 to 0.1 during the first 240 epochs and are then held fixed. The target mean token keep ratio and direction count in Eq. 27 are set to $\rho = 0.5$ and $\kappa = 2$, respectively; $K_{\min} = \max(16, \lceil 0.05HW \rceil)$ prevents an empty or unstable scan. Loss weights are $\eta_t = 10^{-2}$, $\eta_d = 10^{-2}$, and $\eta_b = 5 \times 10^{-3}$. The router-key dimension is 64, the prototype value dimension is 88, the readout dimension is 192, and the EMA decay in Eq. 25 is $\mu = 0.99$. Routing and memory settings are selected on the ImageNet validation split and are shared across all rate points. These defaults are fixed before evaluation on Kodak, CLIC, and Tecnick. The Kodak ablations are post hoc diagnostic evaluations; their results are not used to revise the configuration or select checkpoints. Training and model selection are performed on NVIDIA GeForce RTX 4090 hardware. Prototype initialization uses the 2,000-minibatch reservoir and spherical k-means++ procedure stated in Section 3.3 and is completed before joint optimization.

The Mamba-style projections use expansion factor $E = 2$ and state size $S = 16$. $\mathbf{A} = -\exp(\mathbf{A}_{\log})$ is diagonal and initialized with log-spaced negative time constants; $\mathbf{A}_{\log}$, $\mathbf{D}$, and the input projections are learned. The step bias is initialized so that the softplus step lies in $[10^{-3}, 10^{-1}]$ before budget training. All four scan directions share projection shapes but have independent direction-specific permutations. The spatial context branch is a masked 5×5 convolution and the preceding-group branch is a 1×1 projection; the hyperprior, spatial, and channel features have widths 128, 128, and 64.

At each spatial position within a group, one prototype readout is shared across the 80 channels. The predictor emits $L = 5$ Gaussian-mixture components per channel, giving $80 \times 5 \times 3 = 1{,}200$ outputs per position: a logit, mean, and unconstrained scale for each component. Weights use softmax and scales use $\mathrm{softplus}(\rho) + 10^{-4}$. During arithmetic coding, the mixture CDF is evaluated at half-integer bin boundaries, lower-bounded at 2^{-15}, and quantized to a monotone 16-bit table with total frequency 2^{16}. The first and last table bins are tail/escape bins. The encoder and decoder use the same float32 rounding, clipping, integerization, and tie-breaking rules; a decoded integer must reproduce the same CDF table before a symbol is consumed.

A.6 Complexity and Runtime Measurement Protocol

Training data and evaluation conditions. All learned codecs, including CMIC, DCAE, and ASRIC, are trained on the same ImageNet-1k data. Baselines use their official implementations where available and are evaluated with the weights obtained from this common training dataset. The models retain their respective transforms, entropy models, and coding dependencies. A common wrapper standardizes input preparation, bit accounting, and timing on the same hardware and software stack. Thus, the comparison controls training data and inference conditions without treating the different codecs as architectural ablations. Table 5 summarizes the shared settings.

Test images and bit accounting. The primary complexity and runtime evaluation uses all 41 images in the CLIC Professional Validation set at native resolution, with batch size one on a single NVIDIA GeForce RTX 4090. Each image is reflection-padded to multiples of 64, and the reconstruction is cropped to the original dimensions. The entire padded latent representation is coded. Bitrate is computed from the complete bitstream, including main latents, hyperlatents, and coder headers, and normalized by the original image area. FLOPs and runtime are averaged over images with equal weight. The additional A100 measurements are reported separately in Appendix A.7. VTM-9.1 uses its official intra configuration to provide the BD-Rate reference; it requires no training and is excluded from the learned-model complexity comparison.

Operation counts. Nominal FLOPs sum the neural operations executed during encoding and decoding, including transforms, entropy prediction, and learned routing projections. A multiplication and an addition count as two operations. For ASRIC, only active scan directions and retained-token state updates contribute to the sparse recurrence count. Prototype dot products, weighted aggregation, and readout projections are included. Retrieval is counted once per spatial position within each channel group in each coding pass, while channel-specific parameter prediction is counted separately (Appendix A.4). CMIC and DCAE are profiled as complete codecs, including their respective transforms and entropy predictors.

Index selection, gather, reorder, packing, scatter, and arithmetic coding are not assigned nominal neural FLOPs; their execution time is included in the runtime measurements.

Runtime measurement. For each image, ten warm-up runs precede 50 timed runs. Encoding and decoding are timed separately, with device synchronization at the boundaries of each call. The timed path includes neural transforms, probability prediction, integer-CDF construction, arithmetic coding, and the associated routing and data-movement operations. Dataset loading and file I/O are excluded. We first average the 50 repetitions for each image and then average across the 41 images. This reports end-to-end codec latency under a fixed inference protocol, including costs not represented by the nominal FLOP count.

Table 5: Shared training data and primary complexity-evaluation settings for all learned codecs. A100 timings follow Appendix A.7.

Item	Setting
Training data	Common ImageNet-1k training data
Evaluation set	All 41 CLIC Professional Validation images
Input	Native resolution; reflection padding to multiples of 64
Hardware and batch	One RTX 4090; batch size one
Nominal FLOPs	Executed encoder and decoder neural operations
Timing	10 warm-up runs, then 50 synchronized timed runs per image
Bitrate	Complete bitstream length divided by original image area
Aggregation	Per-image measurements averaged over the test set

A.7 Cross-platform Runtime Validation

Table 6 compares CLIC encoding and decoding times on RTX 4090 and A100 GPUs. For methods evaluated on both devices, input resolution, batch size, padding, and timing follow the same protocol. Timings include neural inference, probability prediction, CDF construction, and arithmetic coding; data loading and file I/O are excluded.

On RTX 4090, ASRIC takes 0.67/0.41 s to encode/decode an image, compared with CMIC's 0.71/0.49 s. On A100, ASRIC takes 0.75/0.35 s versus CMIC's 0.64/0.42 s: decoding is faster, whereas encoding is slower. ASRIC decodes in 0.41 s on RTX 4090 and 0.35 s on A100, versus 0.58 s and 0.54 s for MambaIC. The decoding advantage over the dense SSM baseline persists across both platforms. GLIC remains faster to decode on both devices, so these results do not imply a latency advantage over every architecture.

Table 6: Runtime comparison on CLIC with batch size one. Both devices use the same input preparation and timing protocol.

Method	RTX 4090		A100	
	Enc. (s)	Dec. (s)	Enc. (s)	Dec. (s)
Cheng2020-attn	5.31	> 10	5.22	> 10
TCM-L	0.54	0.49	0.43	0.47
FTIC	> 10	> 10	> 10	> 10
HPCM	0.46	0.45	0.38	0.38
DCAE	0.51	0.49	0.44	0.41
GLIC	0.53	0.36	0.43	0.33
MambaIC	1.17	0.58	1.11	0.54
CMIC	0.71	0.49	0.64	0.42
ASRIC (Ours)	0.67	0.41	0.75	0.35

A.8 KNOWLEDGE-BASE CAPACITY ANALYSIS

We examine sensitivity to the number of KBs while holding the number of prototypes per KB fixed. The eight-KB default is selected on ImageNet validation data before this Kodak evaluation. Table 7 reports a post hoc capacity analysis: BD-Rate improves from four to eight KBs, whereas ten and twelve KBs provide no further RD improvement. All variants require 2.29 TFLOPs at the reported precision under the same Top-2 read budget; the comparison therefore assesses KB count at the same nominal compute cost.

Table 7: Effect of knowledge-base capacity on Kodak. Each KB contains 16,384 prototype values with Top-2 routing. FLOPs are CLIC per-image averages.

# KBs	FLOPs (T)	BD-Rate (%)
4	2.29	-18.49
6	2.29	-18.86
8	2.29	**-19.15**
10	2.29	-19.04
12	2.29	-18.92

A.9 ADDITIONAL DIAGNOSTIC ABLATIONS

These post hoc ablations examine token selection, spatial transitions, mask training, knowledge capacity, retrieval sparsity, and prototype updates around the configuration selected on ImageNet validation data. PSNR BD-Rate is evaluated on Kodak, while FLOPs and decoding time are measured on CLIC following Appendix A.6. Unless otherwise specified, all experiments use the default configuration.

ASRM design choices. We first study the impact of different routing strategies and sparse execution choices in ASRM. The effects of token selection policy, spatial transition modeling, and mask training are summarized in Table 8.

Table 8: Selector, spatial-step, and mask-training ablations. Within each control group, the best Kodak BD-Rate is emphasized.

Control	Variant	FLOPs (T)	Dec. (s)	Kodak BD-Rate (%)
Token policy	Random selection	2.31	0.39	-17.68
	Uniform spatial grid	2.27	0.38	-18.02
	Feature-magnitude ranking	2.34	0.40	-18.47
	Fixed Top-K_{t} TIE	2.30	0.40	-18.83
	Dynamic-threshold TIE	2.29	0.41	**-19.15**
Spatial step	Unit step after compaction	2.27	0.40	-18.76
	Coordinate-aware interval	2.29	0.41	**-19.15**
Mask training	Soft mask with dense execution	5.18	0.47	**-19.21**
	Inference-only hardening	2.28	0.40	-18.53
	Hard-forward straight-through	2.29	0.41	-19.15

Table 8 reports a 0.32-point improvement over fixed Top-K_{t} TIE and a 0.39-point improvement over unit intervals. The nominal costs are close but the table does not verify identical token counts per image and layer. Feature-magnitude ranking is a heuristic control, not a measurement of input texture complexity. Hard-forward training is 0.06 points worse than soft-mask dense execution while reducing FLOPs from 5.18 to 2.29 T. No token-retention correlation with local complexity or reconstruction error is reported; the routing visualization does not supply such a quantitative result.

Knowledge capacity and retrieval sparsity. We next investigate the impact of memory capacity and retrieval density in MASKR. Table 9 reports the compression performance under different prototype capacities and retrieval budgets.

Table 9: Knowledge-capacity and retrieval-sparsity ablations. Each KB contains the stated number of prototypes; the default uses eight KBs and Top-2 routing.

Control	Variant	Total prototypes	FLOPs (T)	Kodak BD-Rate (%)
Capacity/KB	4,096 prototypes	32,768	2.24	-18.31
	8,192 prototypes	65,536	2.26	-18.78
	16,384 prototypes	131,072	2.29	**-19.15**
	32,768 prototypes	262,144	2.35	-19.12
Retrieval	Top-1 (12.5%)	131,072	2.23	-18.72
	Top-2 (25.0%)	131,072	2.29	-19.15
	Top-4 (50.0%)	131,072	2.41	-19.18
	Dense (100%)	131,072	2.73	**-19.20**

Table 9 shows that increasing capacity from 4,096 to 16,384 prototypes per KB improves Kodak BD-Rate, while doubling it to 32,768 provides no further improvement. Dense retrieval improves BD-Rate by 0.05 percentage points over Top-2 at a cost of 0.44 additional TFLOPs. This post hoc comparison quantifies the trade-off of the validation-selected 16,384-prototype, Top-2 configuration.

Prototype update strategy. Finally, we compare prototype update schemes and a multilayer perceptron (MLP) replacement for MASKR within the ASRIC architecture. The results are summarized in Table 10.

Table 10: Prototype-update and predictor ablations on Kodak. The best BD-Rate within each control group is emphasized.

Control	Variant	FLOPs (T)	Kodak BD-Rate (%)
Memory update	Frozen random memory	2.29	-17.91
	Gradient-updated prototypes	2.29	-18.84
	EMA, $\mu = 0.90$	2.29	-18.97
	EMA, $\mu = 0.99$	2.29	**-19.15**
	EMA, $\mu = 0.999$	2.29	-19.02
Predictor	MLP replacement	2.27	-18.51
	MASKR	2.29	**-19.15**

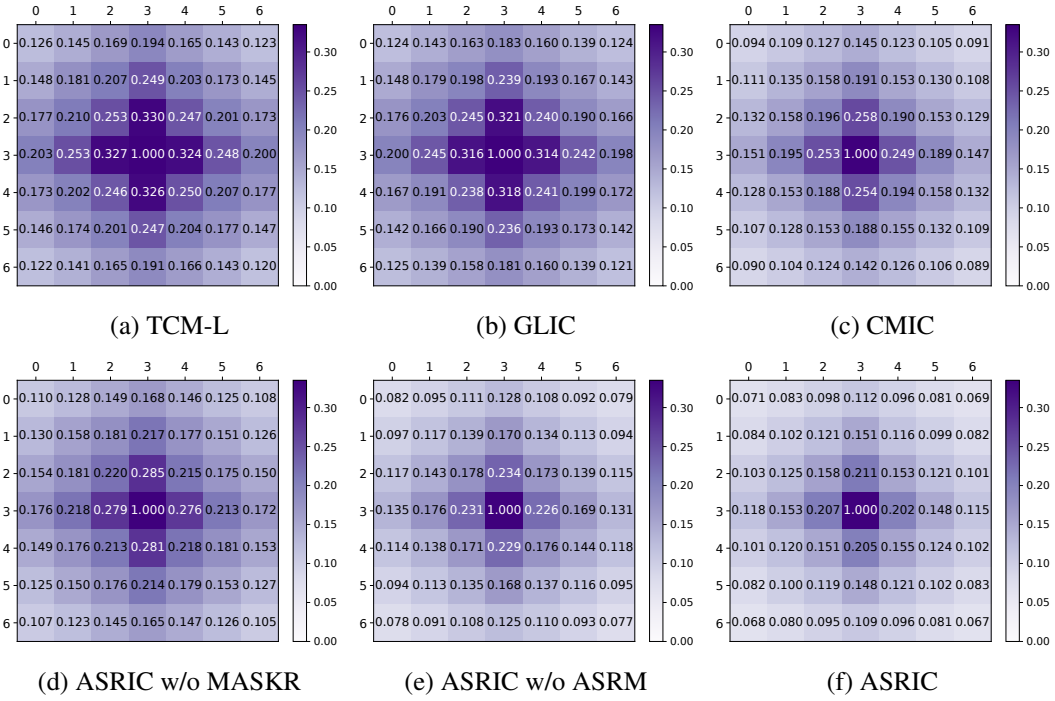
Table 10 reports the lowest BD-Rate for EMA with $\mu = 0.99$ among the evaluated update schemes. Replacing the MLP predictor with MASKR improves the Kodak score by 0.64 percentage points, with 2.29 versus 2.27 TFLOPs. These controls assess entropy-model choices within ASRIC. The DCAE comparison in Table 1 evaluates complete codecs with different transforms and entropy models and therefore does not isolate adaptive bank selection alone.

A.10 Spatial Correlation of Standardized Latents

Spatial dependence in standardized latents provides a diagnostic of structure left unexplained by an entropy model. For a quantized latent $\hat{y}$, the diagnostic centers and scales the latent by the predictive mean μ and standard deviation σ:

$$r = \frac{\hat{y} - \mu}{\sigma}. \tag{30}$$

For a mixture predictor, μ and σ refer to the moments of the mixture distribution. Correlation magnitudes closer to zero at nonzero spatial offsets indicate weaker residual linear dependence. The central self-correlation is one by construction and is excluded when comparing residual dependence.



(a) TCM-L (b) GLIC (c) CMIC

(d) ASRIC w/o MASKR (e) ASRIC w/o ASRM (f) ASRIC

Figure 7: Spatial correlation maps of standardized latents evaluated at $\lambda = 0.0130$. Each panel shows a 7×7 neighborhood centered at index $(3, 3)$. All panels share the same color scale; the central value of 1.000 exceeds its upper limit. Lower off-center correlation magnitudes indicate weaker residual linear spatial dependence.

Figure 7 compares TCM-L, GLIC, CMIC, the two ASRIC ablations, and the full model at $\lambda = 0.0130$. ASRIC consistently exhibits lower off-center correlations than TCM-L, GLIC, and CMIC. Removing either MASKR or ASRM increases the remaining correlations. The two ablations support complementary contributions from knowledge retrieval and adaptive transform computation to reducing residual spatial dependence. In particular, the lower correlations of the full model indicate that its conditional predictions account more effectively for the spatial structure remaining in the quantized latents.

These results provide evidence of reduced residual linear redundancy, consistent with the RD gains in Section 4.2. Spatial correlation is a second-order diagnostic: it does not establish full independence or probability calibration, and its reduction alone does not quantify bitrate savings. The measured RD results establish the corresponding coding benefit.

A.11 Routing Visualization Protocol

The transform-side visualization is produced with the middle-rate checkpoint ($\lambda = 0.013$) on Kodak `kodim22` at its native 768×512 resolution. TIE scores and the corresponding hard mask are exported from the final SAM block at the $H/16 \times W/16$ stage, yielding a 32×48 token grid. No resizing is applied before inference. For display only, each score and mask entry is expanded to its input-space support by nearest-neighbor replication. Consequently, the displayed selection is exactly the hard mask executed by the sparse scan rather than a separately thresholded saliency map.

The entropy-side heatmaps are aggregated over all 24 Kodak images rather than over manually selected crops. For the input-space patch $\mathcal{P}_i$ corresponding to latent location i, local complexity is measured independently of the router by high-frequency energy:

$$e_i = \frac{1}{|\mathcal{P}_i|} \sum_{u \in \mathcal{P}_i} \|\mathbf{x}_u - (G_\sigma * \mathbf{x})_u\|_2^2, \qquad \sigma = 1.5. \tag{31}$$

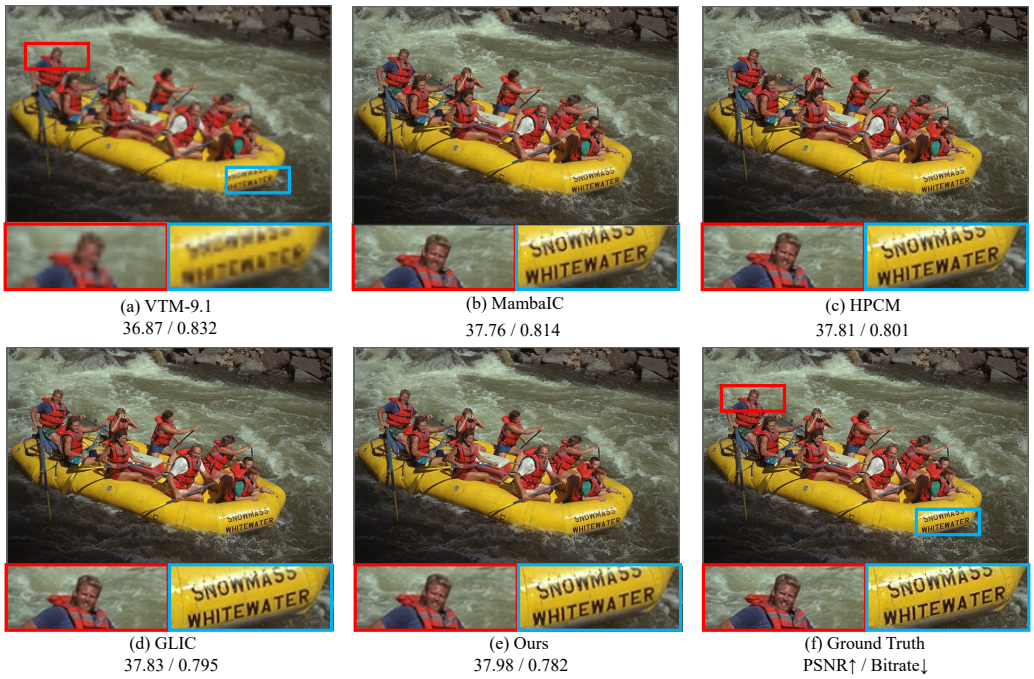
Here G_σ is a Gaussian smoothing kernel with standard deviation σ, and $*$ denotes spatial convolution. Global 33% and 67% quantiles computed over all Kodak locations define the low-, medium-, and high-complexity groups. These thresholds are fixed before KB statistics are accumulated. For channel group k, KB b, and complexity bin q, the normalized Top-K activation ratio is:

$$A_{k,b}^{(q)} = \frac{1}{K N_{k,q}} \sum_{i \in q} \mathbb{I}\Big[b \in \mathrm{TopK}(\mathbf{a}_i^{(k)})\Big], \qquad \sum_{b=1}^{N_{\mathrm{KB}}} A_{k,b}^{(q)} = 1, \tag{32}$$

where $\mathbf{a}_i^{(k)}$ contains the bank logits defined in Section 3.3, with the group superscript restored, and $N_{k,q}$ is the number of spatial positions in channel group k and complexity bin q. Each position contributes one Top-K decision; sharing the readout across channels does not multiply its activation count. The objective partition and dataset-level aggregation prevent the visualization from depending on hand-picked "smooth," "boundary," or "texture" regions.

A.12 Qualitative Reconstruction Comparison

Figure 8 complements the dataset-level rate–distortion results with Kodak image `kodim14`, which contains turbulent water, small faces, and high-contrast lettering. The same image extent and enlarged regions are used for every method. Because the available operating points are close but not exactly rate matched, both PSNR and bitrate are reported below each reconstruction. The comparison is therefore used as qualitative evidence and not as a substitute for the aggregate BD-Rate evaluation.



(a) VTM-9.1
36.87 / 0.832

(b) MambaIC
37.76 / 0.814

(c) HPCM
37.81 / 0.801

(d) GLIC
37.83 / 0.795

(e) Ours
37.98 / 0.782

(f) Ground Truth
PSNR↑ / Bitrate↓

Figure 8: Qualitative reconstruction comparison on Kodak `kodim14` at nearby operating points: (a) VTM-9.1, (b) MambaIC, (c) HPCM, (d) GLIC, (e) ASRIC ("Ours"), and (f) ground truth. For (a)–(e), values denote PSNR (dB) / bitrate (bpp). Red and cyan boxes mark the enlarged face and lettering regions, respectively.

Pronounced smoothing of the face and lettering is observed for VTM-9.1 in Fig. 8(a). Global structure is recovered at lower rates by the learned codecs, although facial boundaries and letter strokes remain softened in several baselines. At 0.782 bpp, ASRIC in (e) reaches 37.98 dB, giving both the highest PSNR and the lowest bitrate among the five coded reconstructions. In the enlarged regions, the face contour and the spacing between lettering strokes are more clearly retained, without an evident structured artifact in the adjacent water. This observation is consistent with the aggregate RD results, but no general conclusion is drawn from a single image.